MULTIMEDIA UNIVERSITY

STUDENT ID NO

MULTIMEDIA UNIVERSITY

SUPPLEMENTARY EXAMINATION

TRIMESTER 1, 2015/2016

DCS5088 – OBJECT ORIENTED PROGRAMMING

(For DIT Students Only)

18 NOV 2015
9.00 AM – 11.00 AM
(2 HOURS)

INSTRUCTIONS TO STUDENTS

1. This examination paper consists of **11** pages of questions excluding the cover page. There are THREE (3) sections in this paper.

2. **SECTION A:** There are **TEN (10)** TRUE/FALSE questions. Answer **ALL** questions on the answer booklet provided (10 marks).

3. **SECTION B:** There are **TWO (2)** structured questions. Answer **ALL** questions on the answer booklet provided (40 marks).

4. **SECTION C:** There is **ONE (1)** structured question. Answer the question on the answer booklet provided (30 marks).

SECTION A **[10 MARKS]**

*Instruction: Answer **'True'** or **'False'** for all statements below and write your answers on the answer booklet provided.*

1. A properly written comment performs no action in a program. "//" is treated as a multiline comment structure.

2. An object is a programming element that contains data and the procedures that operate on the data.

3. Variable names are classified as identifiers. Example of legal variable names are; discount_rate% and Student ID.

4. The 'bool', 'default', 'for' are the special class of identifiers called keywords that have a predefined meaning in C++.

5. The operator >> **is used for input** and << **is used for output**. Both operators recognized the data type supplied.

6. When an argument is passed into a parameter by reference, only a copy of the argument's value is passed. Changes to the parameter do not affect the original argument.

7. A string in C++ is an array of characters ending in the null character ('\0').

8. Most functions have parameters that provide the means of communicating information between functions.

9. Data encapsulation is a process to delete all unnecessary attributes and remain the necessary attributes to describe an object.

10. Inheritance is the ability to create a new object from an existing one. This supports extensibility and reusability within programs.

Continued.....

SECTION B **[40 MARKS]**

Instruction: Answer all questions and write your answers on the answer booklet provided. You do not have to write a full program for each of the questions below.

QUESTION 1 (20 marks)

1.1 Given the incomplete program, answer the following question:

```
#include <iostream>
using namespace std;
int get_total() ;

int main()
{  int total;

   total = get_total();
   cout<<"\nThe total of all the odd numbers are :"<<total<<endl;
   return 0;
}

// 1.1 Write your answer on your answer booklet
```

Complete the function definition (***int get_total ( )***) at segment labelled *'//1.1'* to do the following:

- Get user input of an odd number.
- Using a *while* loop, repeat the input as long as user does not enter an even number. At the same time, the odd numbers will be accumulated in a total variable.
- Return the total. [6 marks]

[**Note:** Refer to sample input/output given below. The bold and italic items are inputs to the program by user]

Sample input/output screen

```
-Enter an odd number to continue-----
-To stop input, enter an even number-
3
-Enter an odd number to continue-----
-To stop input, enter an even number-
5
```

Continued.....

```
-Enter an odd number to continue-----
-To stop input, enter an even number-
11
-Enter an odd number to continue-----
-To stop input, enter an even number-
2


The total of all the odd numbers are :19
```

1.2 Given the program, answer the following questions :

```
#include <iostream>                                          //line1
using namespace std;                                         //line2
struct Rec                                                   //line3
{  private: int s1, s2;                                      //line4
   public:                                                   //line5
   void setData(a, c)                                        //line6
  {                                                          //line7
     s1 = a; s2 = c;                                         //line8
  }                                                          //line9
  void display()                                             //line10
  {                                                          //line11
    for(int i=0; i < s1; i++)                                //line12
    { for(int j=0; j < s2; j++)                              //line13
          cout<<"*";                                         //line14
     cout<<endl;                                             //line15
    }                                                        //line16
   }                                                         //line17
};                                                           //line18
                                                             //line19
int main()                                                   //line20
{   Rec K;                                                   //line21
    K.setData (5,4);                                         //line22
                                                             //line23
    display();                                               //line24
    return 0;                                                //line25
}                                                            //line26
```

(a) Identify **THREE (3)** lines that contain error. Fix the errors (write down the correct answer). [6 marks]

(b) Assuming the program is error free after the corrections are made, trace the output for this program. [2 marks]

1.3 Given the incomplete program, answer the following questions :

```
#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;
const double PI = 3.14,
     SLICE_SIZE = 14.125;   // Sq. inches in each slice
class Cake
{    double cakeDiameter, cakeRadius, cakeArea, slicesPerCake;

     public:
A ---> void input()
       {    cout << "Enter the cake diameter (in inches): ";
            cin  >> cakeDiameter;
       }
B ---> void calculate()
       {    cakeRadius = cakeDiameter / 2;
            cakeArea = PI * pow(cakeRadius, 2);
            slicesPerCake = cakeArea / SLICE_SIZE;
       }
C ---> double getSlicesPerCake()
       { return slicesPerCake;
       }
};


int main()
{
     Cake C[3];
     for(int i=0; i<3; i++)
     { // 1.3 (b) Write your answer on your answer booklet
     }

     return 0;
}
```

Sample input/output screen

```
Enter the cake diameter (in inches): 15.5
Cut this pizza into 13 slices.
Enter the cake diameter (in inches): 10
Cut this pizza into 6 slices.
Enter the cake diameter (in inches): 8.5
Cut this pizza into 4 slices.
```

[**Note:** Refer to sample input/output given above. The bold and italic items are inputs to the program by user]

Continued.....

(a) The labels (A, B, C) at the above are all pointing to the member functions of the class. Which member function (refer to the program) is an accessor function? (Write down the label is sufficient : example A, B or C) [1 mark]

(b) Complete the segment labelled *'// 1.3 (b)'* to do the following for every array element:

i. Call *input()* [1 mark]

ii. Call *calculate()* [1 mark]

iii. Display the number of slices per cake through function call *getSlicesPerCake()* and display the result to nearest whole number. [3 marks]

QUESTION 2 (20 marks)

2.1 Given the following program, answer the following questions :

```
# #include<iostream>
using namespace std;

// 2.1 (a) Write your answer on your answer booklet

class ProductSales
{  string month[5];
   int qty[5];

   public:
       void setdata(string *m, int *q)
       {  for(int i=0; i<5; i++)
        {  month[i] = m[i];
           qty[i]= q[i];
        }

     }

       ~ProductSales()
       { cout<<"\nThank you"<<endl;
       }

       void display_more(SalesPerson *s)
       { cout<<"Salesperson name :"<<s->name<<endl;
       }

       void display()
       {  int i=0;
          while(i<5)
          {  // 2.1 (b) Write your answer on your answer booklet
```

Continued.....

```
        }
    }

    int get_total()
    {  int total = 0;
       for(int i=0; i<5; i++)
          total = total + qty[i];
       return total;
    }

};

int main()
{  string m[5]= { "Jan", "Feb", "Jun", "Oct", "Dec"};
   int  no[5] = { 4, 10, 0, 6, 3};

   ProductSales S1, S2;
   SalesPerson SS1("Jeff Anderson");

   S1.setdata(m, no);
   S1.display();
   S1.display_more(&SS1);
   cout<<"\nThe total units for the five months is"<<S1.get_total()<<
   " units"<<endl;

   return 0;
}
```

[**Note:** Refer to sample output given below]

Sample output screen

```
Month Jan: ****

Month Feb: **********

Month Jun: No units sold

Month Oct: ******

Month Dec: ***

Salesperson name :Jeff Anderson

The total units for the five months is 23 units

Thank you

Thank you
```

See 2.1 (b)

Continued.....

(a) At segment labelled *'//2.1 (a)'*, write the codes to declare a class named *SalesPerson*:

- Private data member ***name*** (string)
- Public parameterized constructor with a string parameter to set the data member ***name***.
- Declare class ***ProductSales*** as a friend

[5 marks]

(b) At segment labelled *'//2.1 (b)'*, complete the *while* loop in the function to display the output that shows the months and the corresponding number of units (***qty***) which are represented by asterisks (*) characters (see the label 2.1 (b) at the sample output screen). If there are zero units for the month, display "No units sold". [7 marks]

2.2 Given the complete program below, answer the following questions :

```
#include<iostream>
using namespace std;
class EconomyCar
{ protected:
    float engine_size;
    int no_of_wheels;

  public:
       EconomyCar(float en, int no)
       { engine_size = en;
         no_of_wheels = no;
       }

       virtual void display()
       {  cout<<"Engine size\t : "<<engine_size<<"CC"<<endl;
          cout<<"No of wheels\t : "<<no_of_wheels<<endl;
       }
};

class MicroCar: public EconomyCar
{ string modelname;
  char electric;
  int year, no_of_stock;
  float price;

  public:
     void display()
     {  cout<<"Model Name\t : "<<modelname<<endl;
        cout<<"Electric powered : "<<electric<<endl;
        cout<<"Year\t\t : "<<year<<endl;
        cout<<"Price\t\t : RM"<<price<<endl;
        cout<<"Engine size\t : "<<engine_size<<"CC"<<endl;
```

Continued.....

```
        cout<<"No of wheels\t : "<<no_of_wheels<<endl;
        cout<<"No of Stock\t : "<<no_of_stock<<endl;

    }

     MicroCar(string n, char e, int y, float p,  int no, float en,
int no_w ): EconomyCar(en, no_w)
     { modelname = n;
       electric = e;
       year = y;
       price = p;
       no_of_stock = no;
     }

   // 2.2 (c) to complete

};

int main()
{ EconomyCar *p;
  MicroCar mm("Tata Nano", 'N', 2014, 20050, 14, 624, 4);
  p = &mm;
  p->display();

  return 0;
}
```

(a) Identify the derived class. [1 mark]

(b) Trace and write down the output produced by the program [4 marks]

(c) At the segment labelled *'// 2.2 (c)'*, define overloaded operator prefix '++' member function. Increment the data member ***no_of_stock.*** [2 marks]

(d) Change ***display()*** function in the *EconomyCar* class to be a pure virtual function. [1 mark]

Continued.....

SECTION C **[30 MARKS]**

Instruction:

Write a **complete program** that gets user input for purchase of three special items and displays the purchase details for those items.

Create a class called *Purchase*:

- Protected data members:
  - *name* : string
  - *total* : float
  - *tax_amount* : float
- Public member functions:
  - *get_tax_amount()* :- Set the tax_amount to 5% of the *total* if the *total* is greater than RM 200.

Create a class called *Special* [**derived publicly** from class *Purchase*]:

- Private data members
  - *items* : array of string, size 3
  - *sub_total* : array of float, size 3
  - *price* : array of float, size 3
  - *qty* : array of int, size 3
- Public member functions:
  - *Default constructor* : - Assign the *items* array with "Designer Mug", Personal Diary 2015" and "Magic Mirror".
    - Assign the price array with 10.50, 6.00 and 22.30.
    - Using a *for* loop, set value 0 for all elements in *sub_total* array and *qty* array.
  - *setdata()* : - Get user input for name.
    - Using a *for* loop (loop 3 times):
      - Get user input for each quantity of the items into *qty* array.
      - Calculate the *sub_total* for every item
      - Accumulate the *sub_total* into *total*.
  - *display()* : - Display *name* and using a *for* loop, display the data members of *Special* Class (*items*, *price*, *qty*, *sub_total*).
    - Display *total* amount.

Continued.....

- Display the tax amount by calling function *get_tax_amount()*.

[**Note:** Refer to sample input/output given below. The bold and italic items are inputs to the program by user]

In ***main()***:

- Declare necessary variables depending on the requirements.
- Get user input for the number of special purchases.
- Create pointer *S* of class *Special.*
- Use the pointer *S* to create a dynamic array of *Special* (the size of the array will be the number of special purchases entered by the user earlier).
- In a *do-while* loop that loop through the array, using pointer *S*:
  - Call *setdata()*.
  - Call *display()*.
- Deallocate memory of the dynamic array.

Sample output screen

```
Please enter how many special purchases :3

Enter name       :Johannes
Enter quantity that you would like to order for [Designer Mug] :10
Enter quantity that you would like to order for [Personal Diary 2015] :2
Enter quantity that you would like to order for [Magic Mirror] :12

----------------------Purchase Details----------------------------
Name     : Johannes
Here are the subtotals for every item
                      Items   Price(RM)    quantity      Subtotal(RM)
                     -------  ----------  ----------    --------------
                Designer Mug       10.50          10            105.00
           Personal Diary 2015        6.00           2             12.00
                Magic Mirror       22.30          12            267.60
Total is RM 384.60
Tax amount (to be absorbed by company) is RM 19.23

Enter name       :Amelia Ooi
Enter quantity that you would like to order for [Designer Mug] :4
Enter quantity that you would like to order for [Personal Diary 2015] :1
Enter quantity that you would like to order for [Magic Mirror] :6

----------------------Purchase Details----------------------------
Name     : Amelia Ooi
Here are the subtotals for every item
                      Items   Price(RM)    quantity      Subtotal(RM)
                     -------  ----------  ----------    --------------
                Designer Mug       10.50           4             42.00
           Personal Diary 2015        6.00           1              6.00
                Magic Mirror       22.30           6            133.80
Total is RM 181.80
Tax amount (to be absorbed by company) is RM 0.00

Enter name       :Fazli Bin Johari
Enter quantity that you would like to order for [Designer Mug] :22
Enter quantity that you would like to order for [Personal Diary 2015] :10
Enter quantity that you would like to order for [Magic Mirror] :6
```

Continued.....

```
----------------------Purchase Details----------------------------
Name      : Fazli Bin Johari
Here are the subtotals for every item
                        Items   Price(RM)     quantity       Subtotal(RM)
                        -------  ----------  ----------     --------------
                  Designer Mug       10.50          22             231.00
             Personal Diary 2015      6.00          10              60.00
                  Magic Mirror       22.30           6             133.80
Total is RM 424.80
Tax amount (to be absorbed by company) is RM 21.24
```

End of Page.